再帰関数Ⅳ

０．目次

１．再帰曲線

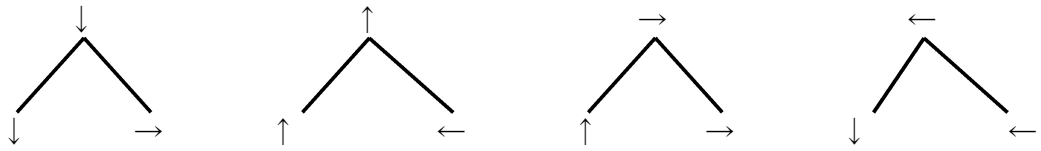
１．１　ドラゴン曲線

ドラゴン曲線は、つぎのルールによって描かれる曲線である。

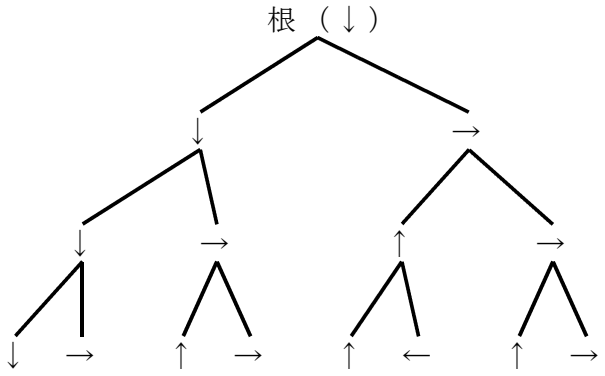
（１）初期状態　↓
（２）変換規則　↓ --> ↓→
　　↑ --> ↑←
　　→ --> ↑→
　　← --> ↓←

［例］１次のドラゴン曲線：↓→
　　２次のドラゴン曲線：↓→↑→
　　３次のドラゴン曲線：↓→↑→↑←↑→

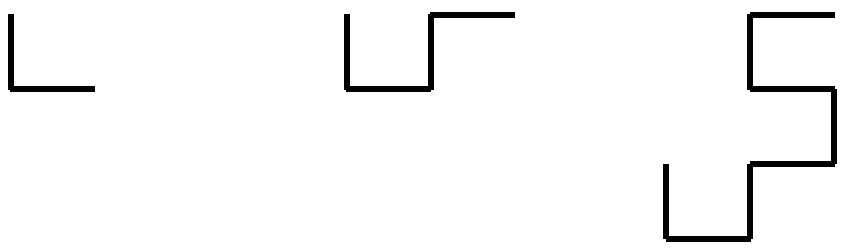
初期状態（↓）につぎつぎ変換を適用していく様子は、木として表せる。



3次のドラゴン曲線に対応する木は次のようになる。



1次のドラゴン曲線　　2次のドラゴン曲線　3次のドラゴン曲線



●プログラム（rc111.c）

```
 1  /* << rc111.c > */
 2  #include <stdio.h>
 3  void dragon(int n, char ch);
 4
 5  int main() {
 6    int n; /* 次数。*/
 7
 8    while( 1 ) {
 9      /* 次数nの読み込み。*/
10      scanf("%d",&n);
11      if( n <= 0 ) { break; }
12      printf("%d次のドラゴン曲線：",n);
13
14      /*ドラゴン曲線。*/
15      dragon(n,'d');
16      printf("¥n");
17    }
18  }
19
20  /* n次のドラゴン曲線。*/
21  void dragon(int n,char ch) {
22    if( n == 0 ) {
23      switch(ch) {
24        case 'u': printf("↑"); break;
25        case 'd': printf("↓"); break;
26        case 'l': printf("←"); break;
27        case 'r': printf("→"); break;
28      }
29    } else {
30      if( ch == 'd' ) { dragon(n-1,'d'); dragon(n-1,'r'); }
31      if( ch == 'u' ) { dragon(n-1,'u'); dragon(n-1,'l'); }
32      if( ch == 'r' ) { dragon(n-1,'u'); dragon(n-1,'r'); }
33      if( ch == 'l' ) { dragon(n-1,'d'); dragon(n-1,'l'); }
34    }
35  }
```

実行結果

```
% cc rc111.c
% ./a.out
1
1次のドラゴン曲線：↓→
2
2次のドラゴン曲線：↓→↑→
3
3次のドラゴン曲線：↓→↑→↑←↑→
4
4次のドラゴン曲線：↓→↑→↑←↑→↑←↓←↑←↑→
0
```

●出力を(x,y)という座標で出力し、Excelを使って図形として表示する。

プログラム(rc121.c)

```
 1  /* << rc121.c >> */
 2  #include <stdio.h>
 3  int x,y,d; /* x:現在のx座標,y:現在のy座標,d:移動距離。 */
 4  void dragon(int n, char ch);
 5
 6  int main() {
 7    int n; /* 次数。 */
 8
 9    /* 次数nの読み込み。 */
10    scanf("%d",&n);
11    if( n <= 0 ) { break; }
12    printf("%d次のドラゴン曲線：",n);
13
14    /* 初期値設定。 */
15    x = 0; y = 0; d = 1;
16
17    /*ドラゴン曲線。 */
18    printf("%d %d¥n",x,y);
19    dragon(n,'d');
20    printf("¥n");
21  }
22  void dragon(int n, char ch) { /* n次のドラゴン曲線。 */
23    if( n == 0 ) {
24      switch(ch) {
25        case 'u':/*↑*/ y = y + d; printf("%d %d¥n",x,y); break;
26        case 'd':/*↓*/ y = y - d; printf("%d %d¥n",x,y); break;
27        case 'l':/*←*/ x = x - d; printf("%d %d¥n",x,y); break;
28        case 'r':/*→*/ x = x + d; printf("%d %d¥n",x,y); break;
29      }
30    } else {
31      switch(ch) {
32        case 'd': dragon(n-1,'d'); dragon(n-1,'r'); break;
33        case 'u': dragon(n-1,'u'); dragon(n-1,'l'); break;
34        case 'r': dragon(n-1,'u'); dragon(n-1,'r'); break;
35        case 'l': dragon(n-1,'d'); dragon(n-1,'l'); break;
36      }
37    }
38  }
```

実行結果

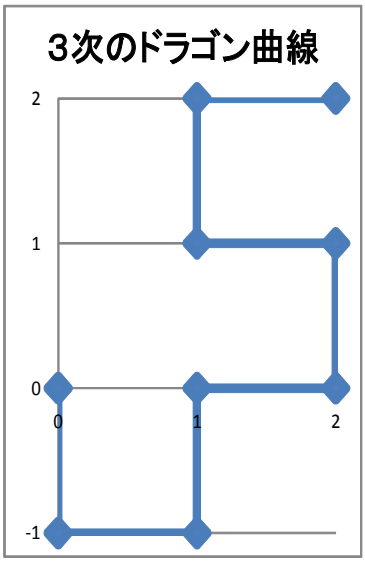
```
% cc rc121.c
% ./a.out
3
3次のドラゴン曲線
0 0
0 -1
1 -1
1 0
2 0
2 1
1 1
1 2
2 2
```

（手順１）(x,y)の組からなるデータファイルを作成する。

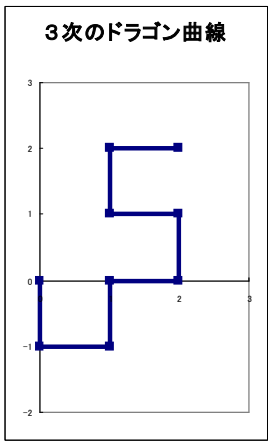
```
% cc rc121.c
% a.out > rc121.rst
3
```

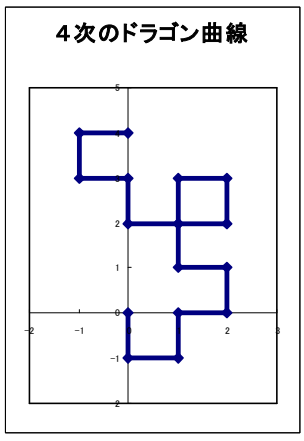
（手順２）Excelのグラフ機能（散布図）を使って表示する。

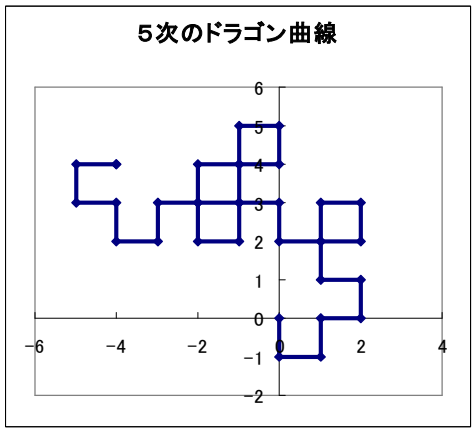
3次のドラゴン曲線	
x座標	y座標
0	0
0	-1
1	-1
1	0
2	0
2	1
1	1
1	2
2	2

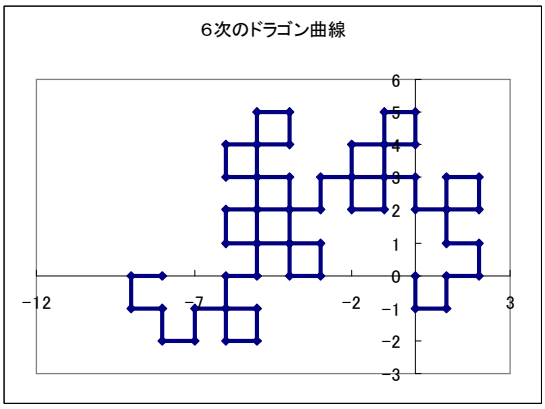


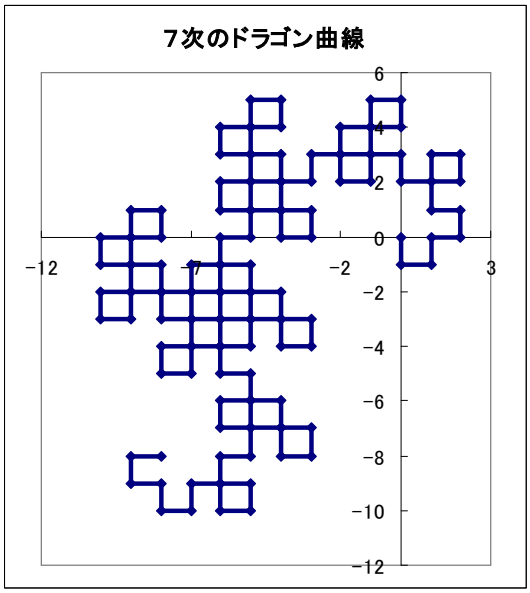
● ３次から８次のドラゴン曲線

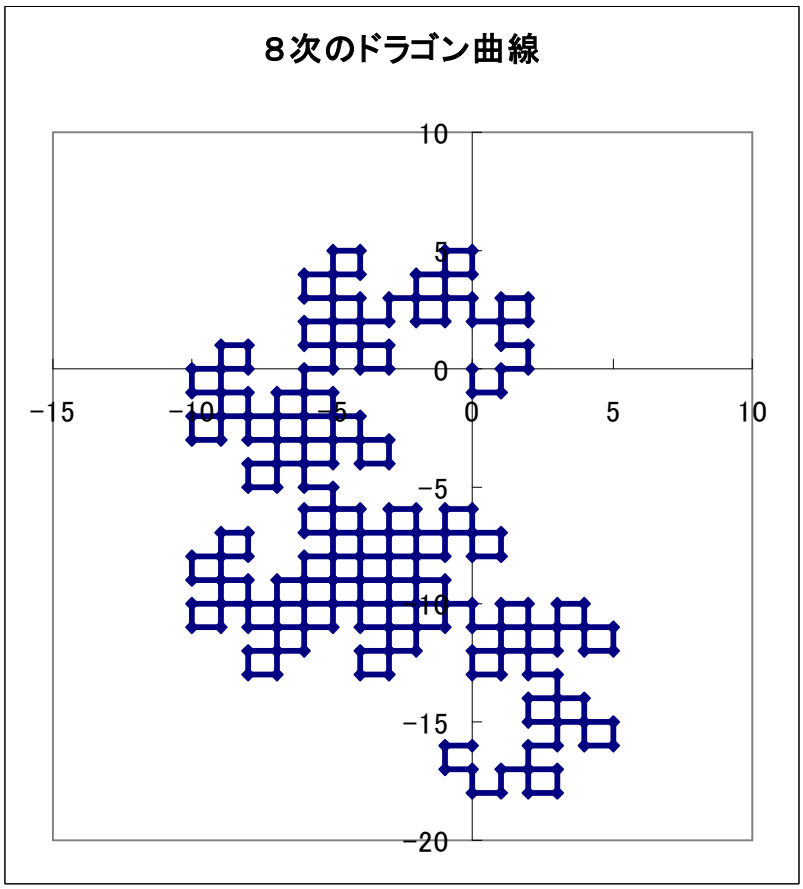







6次のドラゴン曲線
6
5
4
3
2
1
0
−1
−2
−3
−12
−7
−2
3


7次のドラゴン曲線
6
4
2
0
−2
−4
−6
−8
−10
−12
−12
−7
−2
3

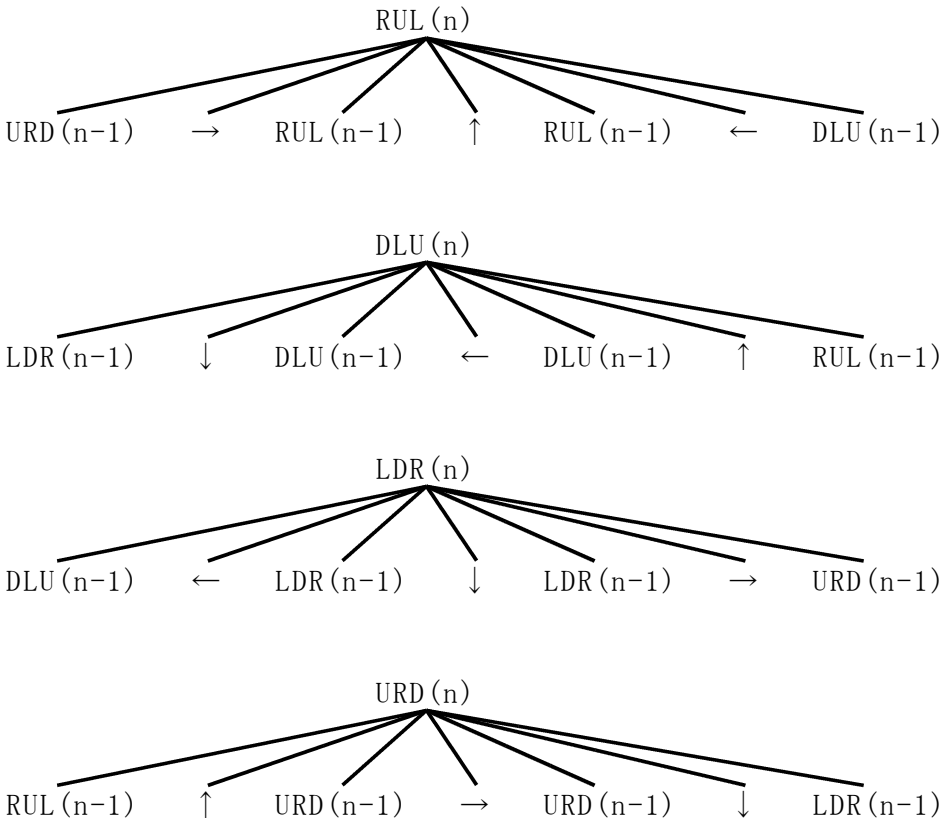

8次のドラゴン曲線
10
5
0
−5
−10
−15
−20
−15
−10
−5
0
5
10

１．２　ヒルベルト曲線

つぎのルールによって描かれる曲線（RUL(n),DLU(n),LDR(n),URD(n)）をヒルベルト曲線という。

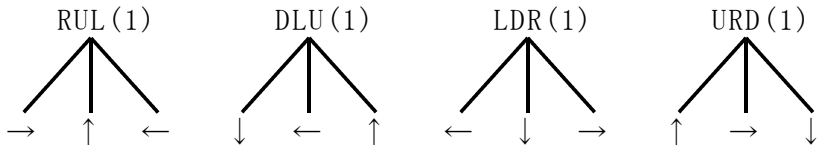
RUL(n) = URD(n-1)→RUL(n-1)↑RUL(n-1)←DLU(n-1)　(n≧1)
　　= ″″　(n=0)
DLU(n) = LDR(n-1)↓DLU(n-1)←DLU(n-1)↑RUL(n-1)　(n≧1)
　　= ″″　(n=0)
LDR(n) = DLU(n-1)←LDR(n-1)↓LDR(n-1)→URD(n-1)　(n≧1)
　　= ″″　(n=0)
URD(n) = RUL(n-1)↑URD(n-1)→URD(n-1)↓LDR(n-1)　(n≧1)
　　= ″″　(n=0)

つぎつぎ変換を適用していく様子は、木として表せる。

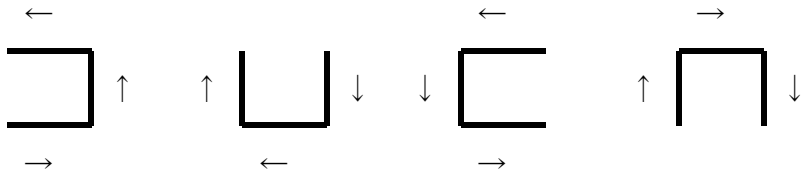


●位数1のヒルベルト曲線（RUL(1),DLU(1),LDR(1),URD(1)）

RUL(1) = URD(0)→RUL(0)↑RUL(0)←DLU(0) = →↑←
DLU(1) = LDR(0)↓DLU(0)←DLU(0)↑RUL(0) = ↓←↑
LDR(1) = DLU(0)←LDR(0)↓LDR(0)→URD(0) = ←↓→
URD(1) = RUL(0)↑URD(0)→URD(0)↓LDR(0) = ↑→↓



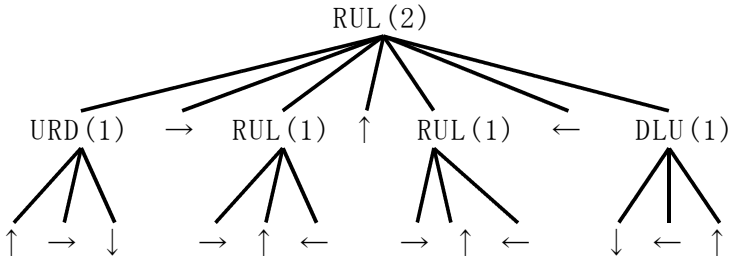
対応する位数1のヒルベルト曲線を示す。



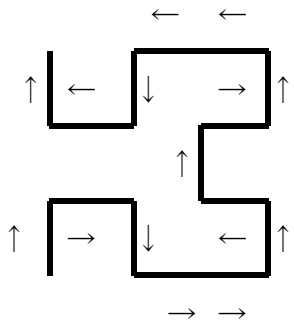
●位数2のひとつのヒルベルト曲線（RUL(2)）

RUL(2) = URD(1) → RUL(1) ↑ RUL(1) ← DLU(1)
= {RUL(0)↑URD(0)→URD(0)↓LDR(0)} →
{URD(0)→RUL(0)↑RUL(0)←DLU(0)} ↑
{URD(0)→RUL(0)↑RUL(0)←DLU(0)} ←
{LDR(0)↓DLU(0)←DLU(0)↑RUL(0)}
= ↑→↓→→↑←↑→↑←←↓←↑

位数2のヒルベルト曲線（RUL(2)）を表現する木は次のようになる。



この木を根(RUL(2))から出発し、反時計回りにたどることによって得られる葉の列(↑→↓→→↑←↑→↑←←↓←↑)が位数2のヒルベルト曲線に対応する。



●出力を(x,y)という座標で出力し、Excelのグラフ機能を使って図形として表示する。

（手順１）(x,y)の組からなるデータファイルを作成する。

```
% cc rc121.c
% a.out > rc121.rst
3
```

（手順２）Excelで表示する。

●プログラム(rc121.c)

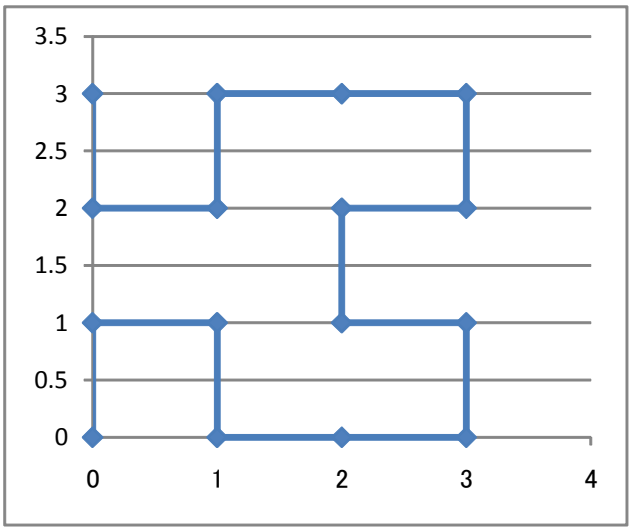
```
1  /* << rc121.c >> */
2  #include <stdio.h>
3  #include <stdlib.h>
4  int x,y; /* x:現在のｘ座標、y:現在のｙ座標。*/
5  void RUL(int n),DLU(int n),LDR(int n),URD(int n);
6
7  int main() {
8    int n; /* 位数。*/
9
10   /* 位数の読み込み。*/
11   scanf("%d",&n);
12   if( n <= 0 ) { exit(0); }
13
14   /* 初期設定。*/
15   x = 0; y = 0;
16   printf("%5d %5d¥n",x,y);
17
18   /* ヒルベルト曲線。*/
19   printf("位数%dのヒルベルト曲線¥n",n);
20   RUL(n);
21   printf("¥n");
22 }
23
24 void RUL(int n) {
25   if( n <= 0 ) { return; }
26   URD(n-1); x = x + 1; printf("%5d %5d¥n",x,y);
27   RUL(n-1); y = y + 1; printf("%5d %5d¥n",x,y);
28   RUL(n-1); x = x - 1; printf("%5d %5d¥n",x,y);
29   DLU(n-1);
30 }
31
32 void DLU(int n) {
33   if( n <= 0 ) { return; }
34   LDR(n-1); y = y - 1; printf("%5d %5d¥n",x,y);
35   DLU(n-1); x = x - 1; printf("%5d %5d¥n",x,y);
36   DLU(n-1); y = y + 1; printf("%5d %5d¥n",x,y);
37   RUL(n-1);
38 }
39
40 void LDR(int n) {
41   if( n <= 0 ) { return; }
42   DLU(n-1); x = x - 1; printf("%5d %5d¥n",x,y);
43   LDR(n-1); y = y - 1; printf("%5d %5d¥n",x,y);
44   LDR(n-1); x = x + 1; printf("%5d %5d¥n",x,y);
45   URD(n-1);
```

```
46  }
47
48  void URD(int n) {
49    if( n <= 0 ) { return; }
50    RUL(n-1); y = y + 1; printf("%5d %5d¥n",x,y);
51    URD(n-1); x = x + 1; printf("%5d %5d¥n",x,y);
52    URD(n-1); y = y - 1; printf("%5d %5d¥n",x,y);
53    LDR(n-1);
    }
```

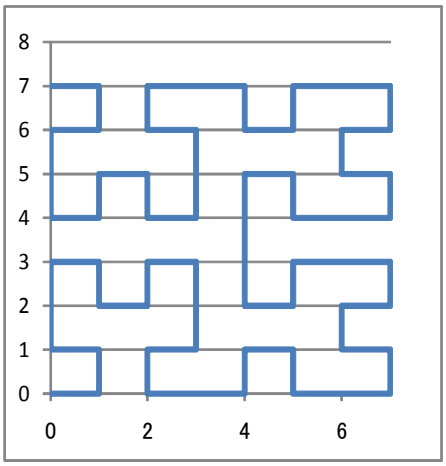
実行結果

```
% cc rc121.c
% a.out
2
位数2のヒルベルト曲線
    0    0
    0    1
    1    1
    1    0
    2    0
    3    0
    3    1
    2    1
    2    2
    3    2
    3    3
    2    3
    1    3
    1    2
    0    2
    0    3
```

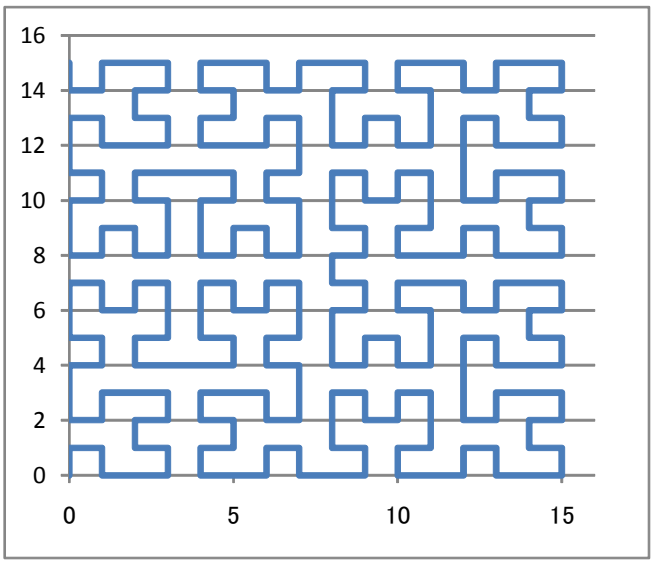
位数２のヒルベルト曲線



位数３のヒルベルト曲線



位数４のヒルベルト曲線



１．３　シェルピンスキー曲線

つぎのルールによって描かれる曲線をシェルピンスキー曲線という。

```
URD(n) = URD(n-1) a LUR(n-1) e RDL(n-1) d URD(n-1)
LUR(n) = LUR(n-1) b DLU(n-1) f URD(n-1) a LUR(n-1)
DLU(n) = DLU(n-1) c RDL(n-1) g LUR(n-1) b DLU(n-1)
RDL(n) = RDL(n-1) d URD(n-1) h DLU(n-1) c RDL(n-1)
初期条件：URD(0)= "" LUR(0) = "" DLU(0) = "" RDL(0) = ""
```

ただし、a,b,c,d,e,f,g,h は、つぎのようなベクトルとする。

```
a = (1,1)   b = (-1,1)   c = (-1,-1)   d = (1,-1)
e = (2,0)   f = (0,2)   g = (-2,0)   h = (0,-2)
```

●位数０のシェルピンスキー曲線：abcd
位数１のシェルピンスキー曲線：aedabfabcgbcdhcd

●出力を(x,y)という座標で出力し、gnuplotを使って図形として表示する。

（手順１）(x,y)の組からなるデータファイルを作成する。

```
% cc rc131.c
% a.out > rc131.rst
3
```

（手順２）Excelで表示する。

●プログラム(rc131.c)

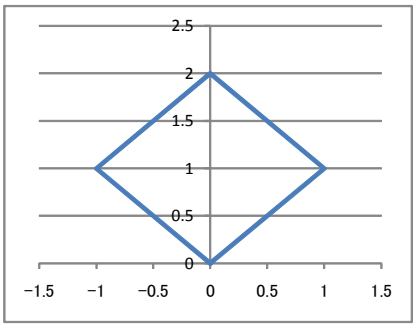
```
1  /* << rc131.c >> */
2  #include <stdio.h>
3  #include <stdlib.h>
4  int x,y; /* x:現在のｘ座標、y:現在のｙ座標。*/
5  void s(int n),URD(int n),LUR(int n),DLU(int n),RDL(int n);
6
7  int main() {
8    int n; /* 位数。*/
9
10   /* 位数nの読み込み。*/
11   scanf("%d",&n);
12   if( n <= 0 ) { exit(0); }
13
14   /* 初期値設定。*/
15   x = 0; y = 0;
16   printf("%5d %5d¥n",x,y);
17
```

```
18    /* シェルピンスキー曲線。*/
19    printf("位数%dのシェルピンスキー曲線¥n",n);
20    s(n);
21    printf("¥n");
22  }
23
24  void URD(int n) {
25    if( n <= 0 ) { return; }
26    URD(n-1); x = x+1; y = y+1; printf("%5d %5d¥n",x,y);
27    LUR(n-1); x = x+2; y = y+0; printf("%5d %5d¥n",x,y);
28    RDL(n-1); x = x+1; y = y-1; printf("%5d %5d¥n",x,y);
29    URD(n-1);
30  }
31
32  void LUR(int n) {
33    if( n <= 0 ) { return; }
34    LUR(n-1); x = x - 1; y = y + 1; printf("%5d %5d¥n",x,y);
35    DLU(n-1); x = x + 0; y = y + 2; printf("%5d %5d¥n",x,y);
36    URD(n-1); x = x + 1; y = y + 1; printf("%5d %5d¥n",x,y);
37    LUR(n-1);
38  }
39
40  void DLU(int n) {
41    if( n <= 0 ) { return; }
42    DLU(n-1); x = x - 1; y = y - 1; printf("%5d %5d¥n",x,y);
43    RDL(n-1); x = x - 2; y = y + 0; printf("%5d %5d¥n",x,y);
44    LUR(n-1); x = x - 1; y = y + 1; printf("%5d %5d¥n",x,y);
45    DLU(n-1);
46  }
47
48  void RDL(int n) {
49    if( n <= 0 ) { return; }
50    RDL(n-1); x = x + 1; y = y - 1; printf("%5d %5d¥n",x,y);
51    URD(n-1); x = x + 0; y = y - 2; printf("%5d %5d¥n",x,y);
52    DLU(n-1); x = x - 1; y = y - 1; printf("%5d %5d¥n",x,y);
53    RDL(n-1);
54  }
55
56  /* 位数 n のシェルピンスキー曲線。*/
57  void s(int n) {
58    URD(n); x = x + 1; y = y + 1; printf("%5d %5d¥n",x,y);
59    LUR(n); x = x - 1; y = y + 1; printf("%5d %5d¥n",x,y);
60    DLU(n); x = x - 1; y = y - 1; printf("%5d %5d¥n",x,y);
61    RDL(n); x = x + 1; y = y - 1; printf("%5d %5d¥n",x,y);
62  }
```

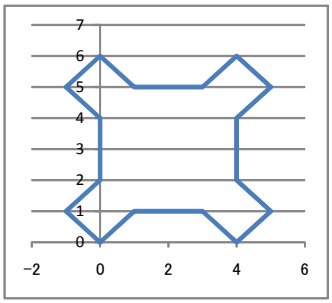
実行結果

```
% cc rc131.c
% ./a.out
1
位数1のシェルピンスキー曲線
     0       0
     1       1
     3       1
     4       0
     5       1
     4       2
     4       4
     5       5
     4       6
     3       5
     1       5
     0       6
    -1       5
     0       4
     0       2
    -1       1
     0       0
```

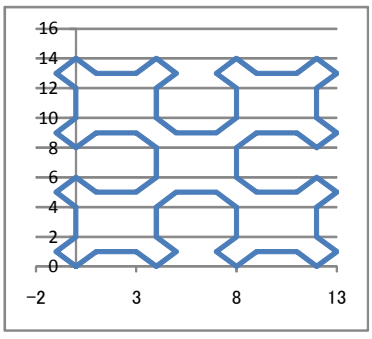
位数０のシェルピンスキー曲線



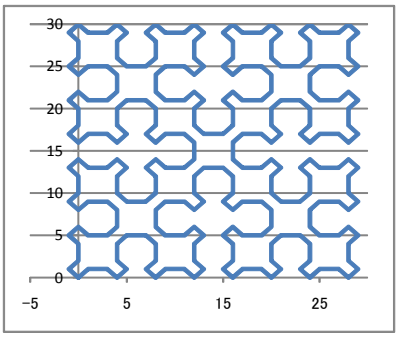
位数１のシェルピンスキー曲線



位数２のシェルピンスキー曲線

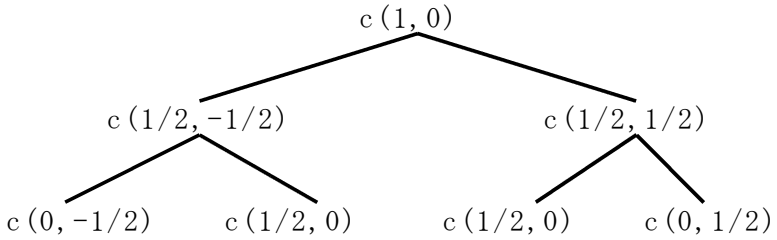


位数３のシェルピンスキー曲線



１．４　Ｃ曲線

現在の点からｘ軸方向にｘ，ｙ軸方向にｙ進む操作をc(x,y)とする。
操作C(x,y)が、操作c((x+y)/2,(-x+y)/2)と操作c((x-y)/2,(x+y)/2)に繰り返し分解されるとき描かれる曲線をＣ曲線という。



●出力を(x,y)という座標で出力し、Excelを使って図形として表示する。

（手順１）(x,y)の組からなるデータファイルを作成する。

```
% cc rc141.c
% a.out > rc141.rst
3
```

（手順２）Excelで表示する。

●プログラム(rc141.c)

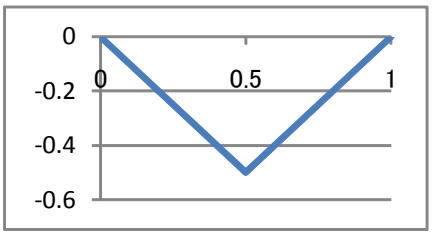
```
 1 /* << rc141.c >> */
 2 #include <stdio.h>
 3 #include <stdlib.h>
 4 float x,y; /* x:現在のｘ座標、y:現在のｙ座標。*/
 5 int n; /* 分解の深さ。*/
 6 void c(int k, float dx, float dy);
 7
 8 int main() {
 9
10   /* 分解の深さnの読み込み。*/
11   scanf("%d",&n);
12   if( n <= 0 ) { exit(0); }
13   printf("分解が%dまでのＣ曲線¥n",n);
14
15   /* 初期値設定。*/
16   x = 0.0; y = 0.0;
17   printf("%6.2f %6.2f¥n",x,y);
18
19   /* C曲線。*/
20   c(0,1.0,0.0);
21   printf("¥n");
22 }
23
24 /* 分解がｋまでのＣ曲線。*/
25 void c(int k,float dx,float dy) {
26   if( k == n ) {
```

```
27      x = x + dx; y = y + dy;
28      printf("%6.2f %6.2f¥n",x,y);
29    } else {
30      c(k+1,(dx+dy)/2,(-dx+dy)/2);
31      c(k+1,(dx-dy)/2,(dx+dy)/2);
32    }
33    return;
34  }
```

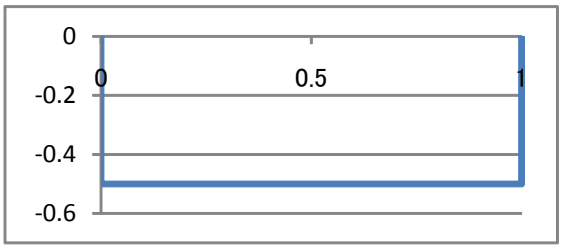
実行結果

```
% cc rc141.c
% ./a.out
1
分解が1までのＣ曲線
   0.00    0.00
   0.50   -0.50
   1.00    0.00
% ./a.out
2
分解が2までのＣ曲線
   0.00    0.00
   0.00   -0.50
   0.50   -0.50
   1.00   -0.50
   1.00    0.00
% ./a.out
3
分解が3までのＣ曲線
   0.00    0.00
  -0.25   -0.25
   0.00   -0.50
   0.25   -0.75
   0.50   -0.50
   0.75   -0.75
   1.00   -0.50
   1.25   -0.25
   1.00    0.00
```

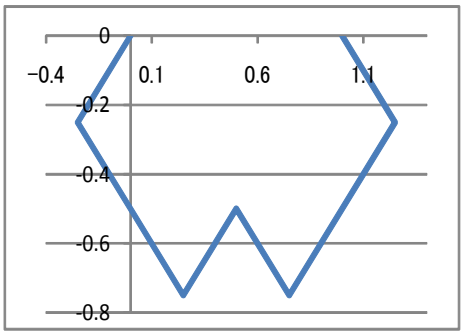
分解が１までのC曲線



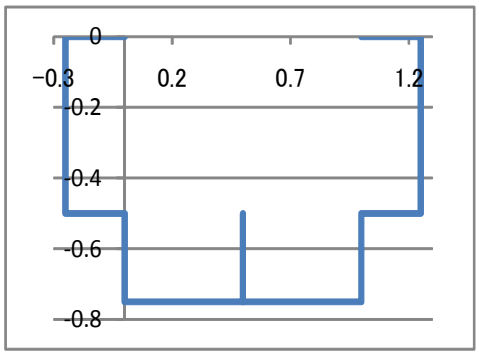
分解が２までのC曲線



分解が３までのC曲線



分解が４までのC曲線



分解が５までのC曲線